

# Poglavlje 1

## Modeliranje krožne topologije P2P v orodju OMNeT++

Stefan Jakjoski, Branko Raičković, Nik Janoš in Tadej Pariš

**Povzetek** V tem seminarju predstavljamo modeliranje in simulacijo krožne (ring) topologije omrežja v orodju OMNeT++. Analiziramo vpliv različnih parametrov na zmogljivost omrežja, s posebnim poudarkom na povprečnem času potovanja paketa od izvornega do ponornega vozlišča. Implementirali smo enostavno P2P krožno omrežje s 7 vozlišči in analizirali delovanje pri različnih obremenitvah (1 Mbps, 30 Mbps, 150 Mbps in 500 Mbps). Rezultati kažejo, da ima hitrost prenosa podatkov ključen vpliv na zakasnitev in zasedenost čakalnih vrst.

### 1.1 Uvod in motivacija

Omrežne topologije predstavljajo temeljni koncept pri načrtovanju računalniških omrežij. Izbira pravilne topologije je ključnega pomena za zagotavljanje učinkovitega prenosa podatkov, minimizacijo zakasnitev in obvladovanje prometnih nasičenj.

V tem seminarju obravnavamo štiri najpogosteje omrežne topologije:

1. **Zvezdasta topologija (Star)** – V omrežju z zvezdasto topologijo je vsaka naprava povezana na skupno centralno vozlišče (hub). Ločimo aktivno in pasivno zvezdasto topologijo. Pri aktivni centralno vozlišče pošlje prejeto sporočilo napravam v omrežju, pri pasivni pa centralno vozlišče služi le kot povezava.

#### Prednosti

- Odpornost do napak
- Eliminira trke paketov
- Cenovno ugodno: vsaka naprava potrebuje le ena vrata za komunikacijo
- Raztegljivo: dodajanje naprave ne spremeni strukture obstoječega omrežja
- Enostavna diagnoza napak

### Slabosti

- Centralno vozlišče je točka odpovedi
  - Zmogljivost odvisna od centralnega vozlišča
2. **Mrežna topologija (Mesh)** – V omrežju z mesh topologijo je vsaka naprava povezana direktno z vsako drugo napravo. Tako omrežje je torej decentralizirano.

### Prednosti

- Več možnih poti omogoča boljšo odpornost do napak: če povezava odpove, uberemo drugo pot
- Raztegljivo (scalable) omrežje: dodajanje naprave ne spremeni strukture obstoječega omrežja
- Učinkovit prenos podatkov: ne potujejo skozi centralno vozlišče

### Slabosti

- Kompleksna izdelava in vzdrževanje: vsaka naprava je povezana z vsako (alternativa: delno mrežasta topologija)
  - Visoka cena
  - Možnost redundantnih povezav
3. **Krožna topologija (Ring)** – Krožna point-to-point topologija (ali topologija obroča) opisuje omrežje, kjer je vsaka naprava povezana z natanko dvema drugima napravama v zaprti zanki. Krožne topologije omogočajo krožni prenos podatkov med  $n$  vozlišči. Podatkovni paketi potujejo v eno smer in prehajajo skozi vsako vozlišče v obroču, dokler ne dosežejo cilja.  
V krožnih topologijah lahko podatke hkrati prenaša le eno vozlišče. To se nadzruje z žetonom, ki kroži po omrežju in omogoča posameznemu vozlišču prenos podatkov. Vozlišče ne sme prenatisati podatkov brez posedovanja žetona.

### Prednosti

- Naenkrat lahko prenaša le eno vozlišče, kar zmanjšuje trke paketov (packet collision)
- Implementacija je pogosto preprosta; kompleksnost dodajanja več vozlišč se linearno skalira, za razliko od kvadratnega skaliranja mrežaste (mesh) topologije

### Slabosti

- Če eno vozlišče odpove, se celotno omrežje izklopi
- Zmogljivost omrežja je enostavno preobremeniti

4. **Spine Leaf topologija** – Je topologija, sestavljena iz dveh slojev stikal: leaf in spine. Uporabljena je v glavnem v podatkovnih centrih, kjer imamo opravka z velikim številom naprav. Je alternativa tradicionalni troslojni arhitekturi. Leaf so stikala, ki prejemajo podatke, spine pa so stikala, ki leaf sloj povezujejo v mrežasto topologijo.

### Prednosti

- Redundanca stikal
- Raztegljivost
- Nizka latenca v primerjavi s tradicionalno

### Slabosti

- Potrebujemo več stikal
- Za veliko uporab prekompleksna topologija

V tem delu podrobno analiziramo **krožno topologijo P2P** in jo implementiramo v orodju OMNeT++. Krožna topologija je posebej primerna za analizo prometnih nasičenj, saj ima deterministično pot prenosa – vsak paket potuje od izvornega do ciljnega vozlišča po krožni poti [1].

## 1.2 Opis protokola

V implementirani krožni topologiji uporabljam enostaven protokol za posredovanje paketov, ki temelji na štetju preskokov (hop count). Protokol deluje na naslednji način:

1. Vsako vozlišče generira pakete z naključnimi ciljnimi vozlišči.
2. Paket vsebuje informacije o izvornem vozlišču, ciljnem vozlišču, številu preskokov in času pošiljanja.
3. Ko vozlišče prejme paket, preveri:
  - Če je trenutno vozlišče cilj paketa in je paket naredil vsaj en preskok, se paket sprejme in zabeleži se zakasnitev.
  - Če se je paket vrnil na izvorno vozlišče, se zavrže (izgubljen paket).
  - Če je presegena omejitev preskokov (hopLimit), se paket zavrže.
4. V nasprotnem primeru se število preskokov poveča in paket se posreduje naslednjemu vozlišču.

Ta protokol je podoben enostavnemu posredovanju v obroču, kjer paketi potujejo samo v eni smeri.

## 1.3 Uporabljene knjižnice in moduli

Za implementacijo krožne topologije smo uporabili naslednje komponente ogrodja OMNeT++ [1]:

### 1.3.1 Osnovni moduli

- cSimpleModule – Osnovni razred za implementacijo aktivnih modulov, ki procesirajo sporočila.
- cPacket – Razred za predstavitev omrežnih paketov z možnostjo nastavljanja velikosti in parametrov.
- cPacketQueue – Čakalna vrsta za shranjevanje paketov pred pošiljanjem.
- cDatarateChannel – Kanal s konfigurirano hitrostjo prenosa podatkov in zakasnitvijo.
- cMessage – Osnovni razred za sporočila, ki jih uporabljam za časovnike.

### 1.3.2 Signali in statistika

- simsignal\_t – Signal za beleženje zakasnitev paketov.

recordScalar() – Funkcija za beleženje skalarnih vrednosti ob koncu simulacije.

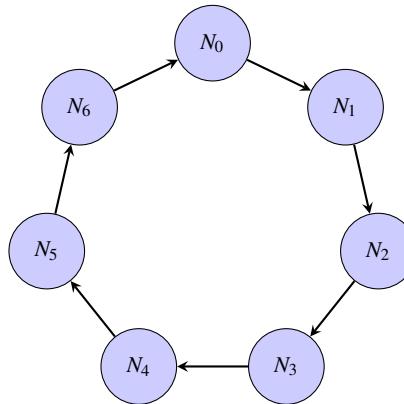
### ***1.3.3 NED jezik***

Za opis topologije omrežja uporabljam NED jezik, ki omogoča deklarativno definicijo modulov, povezav in parametrov.

## 1.4 Rešitev

### 1.4.1 Struktura omrežja

Implementirali smo krožno omrežje s 7 vozlišči, kjer je vsako vozlišče povezano z naslednjim v krogu. Struktura omrežja je prikazana na sliki 1.1.



Slika 1.1: Krožna topologija s 7 vozlišči. Puščice prikazujejo smer prenosa paketov.

### 1.4.2 Definicija omrežja v NED jeziku

Omrežje je definirano v datoteki `ringnetwork.ned`:

```

1 network circularTopology {
2     parameters:
3         int numNodes = default(6);
4         double datarate @unit(bps) = default(10Mbps);
5     submodules:
6         node[numNodes]: Node;
7     connections allowunconnected:
8         for i=0..numNodes-1 {
9             node[i].out++ --> DatarateChannel {
10                 datarate; delay = 1ms;
11             } --> node[(i+1) % numNodes].in++;
12         }
13     }
  
```

Listing 1.1: Definicija krožne topologije v NED jeziku

### 1.4.3 Implementacija vozlišča

Vozlišče je implementirano v razredu Node, ki deduje od cSimpleModule. Glavne funkcionalnosti vključujejo:

- Generiranje paketov z naključnimi ciljnimi vozlišči
- Upravljanje čakalne vrste za pakete
- Posredovanje paketov naslednjemu vozlišču
- Beleženje statistik (zakasnitev, izgubljeni paketi)

Ključni del kode za obdelavo paketov:

```

1 void Node::handleMessage(cMessage *msg) {
2     cPacket *pkt = check_and_cast<cPacket *>(msg);
3     int hops = pkt->par("hops");
4     int dest = pkt->par("destinationId");
5
6     // Preveri, ce je cilj dosezen
7     if (nodeId == dest && hops > 0) {
8         simtime_t delay = simTime() - startTime;
9         emit(delaySignal, delay);
10        delete pkt;
11        return;
12    }
13
14    // Posreduj naprej
15    pkt->par("hops") = hops + 1;
16    forwardPacket(pkt);
17 }
```

Listing 1.2: Obdelava prejetih paketov

## 1.5 Simulacije in rezultati

### 1.5.1 Parametri simulacije

Za analizo zmogljivosti omrežja smo identificirali 5 ključnih parametrov, ki bistveno vplivajo na delovanje:

Tabela 1.1: Parametri simulacije

Parameter	Vrednost	Opis
numNodes	7	Število vozlišč v krogu
hopLimit	10	Maksimalno število preskokov
queueLength	50	Dolžina čakalne vrste
packetSize	1 MB	Velikost paketa
sendInterval	1 s (exp.)	Interval pošiljanja

### 1.5.2 Konfiguracije obremenitve

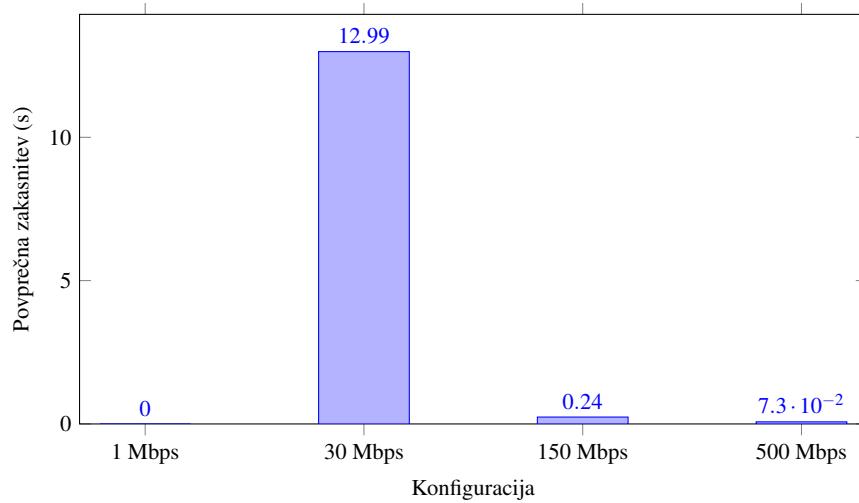
Omrežje smo testirali pri štirih različnih hitrostih prenosa, ki simulirajo različne stopnje obremenitve:

Tabela 1.2: Konfiguracije obremenitve omrežja

Konfiguracija	Hitrost	Obremenitev
config1	1 Mbps	Nadpovprečna (preobremenitev)
config2	30 Mbps	Normalna
config3	150 Mbps	Podpovprečna
config4	500 Mbps	Minimalna

### 1.5.3 Analiza povprečnega časa potovanja paketa

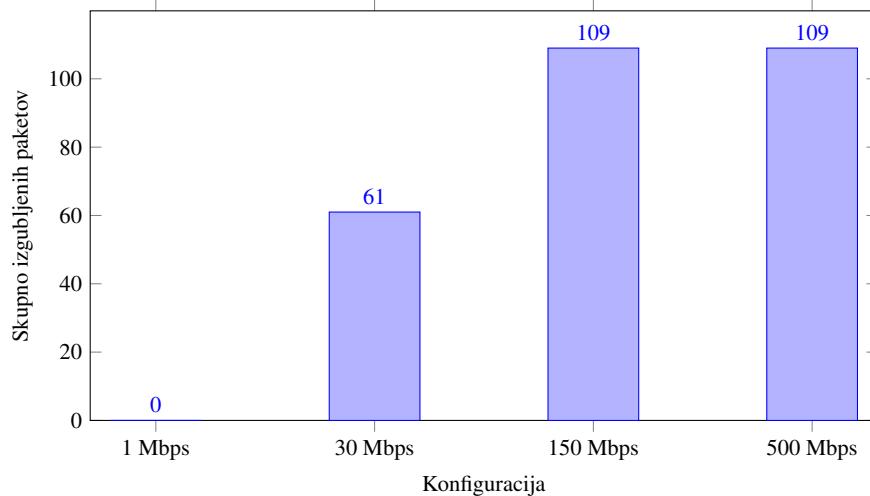
Povprečen čas potovanja paketa od izvornega do ponornega vozlišča (End-to-End Delay) je ključna metrika za oceno zmogljivosti omrežja.



Slika 1.2: Povprečna zakasnitev paketa pri različnih hitrostih prenosa. Pri 1 Mbps omrežje ni sposobno dostaviti paketov (preobremenitev).

Pri konfiguraciji z 1 Mbps je omrežje popolnoma preobremenjeno – čakalne vrste se napolnijo in vsi paketi se zavržejo, preden dosežejo cilj. Pri 30 Mbps opazimo povprečno zakasnitev približno 13 sekund, kar kaže na visoko obremenitev. Pri višjih hitrostih (150 Mbps in 500 Mbps) zakasnitev pada pod 1 sekundo.

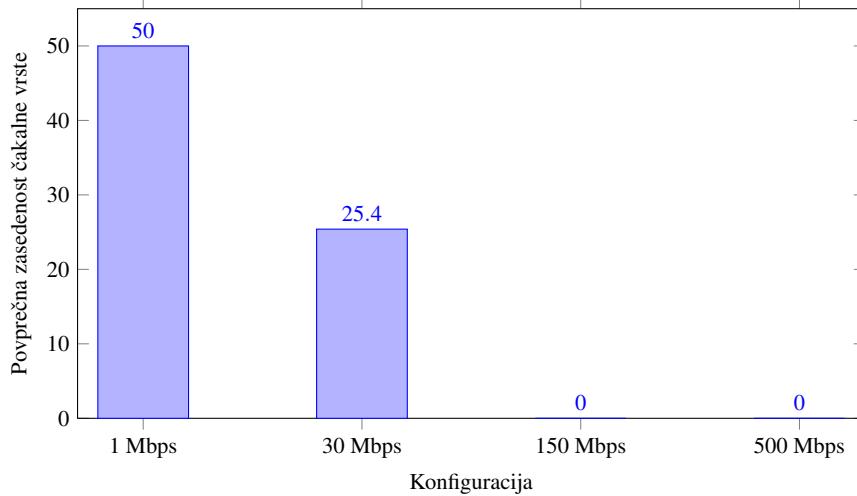
#### 1.5.4 Analiza izgubljenih paketov



Slika 1.3: Število izgubljenih paketov pri različnih konfiguracijah.

Zanimivo opažanje je, da se pri nižji hitrosti (30 Mbps) izgubi manj paketov kot pri višjih hitrostih. To je zato, ker paketi ostanejo v čakalnih vrstah in se počasneje obdelujejo, medtem ko pri višjih hitrostih paketi hitreje krožijo in se vrnejo na izvorno vozlišče (kar šteje kot izguba).

### 1.5.5 Analiza zasedenosti čakalnih vrst



Slika 1.4: Povprečna zasedenost čakalnih vrst ob koncu simulacije.

Pri 1 Mbps so čakalne vrste popolnoma zasedene (50 elementov – maksimalna kapaciteta), kar pomeni, da se novi paketi zavržejo. Pri 30 Mbps je povprečna zasedenost približno 25 elementov, kar kaže na zmerno obremenitev. Pri višjih hitrostih so čakalne vrste prazne, kar pomeni, da se paketi prenašajo hitreje, kot se generirajo.

### 1.5.6 Določitev optimalne kapacitete povezav

Na podlagi rezultatov lahko določimo optimalno kapaciteto povezav za naše omrežje:

- **1 Mbps** – Popolnoma neustrezno. Omrežje je preobremenjeno in ne more dostaviti paketov.
- **30 Mbps** – Mejna ustreznost. Visoka zakasnitev (13 s), a paketi se dostavljajo. Čakalne vrste so delno zasedene.
- **150 Mbps** – Ustrezno. Nizka zakasnitev (0.24 s), prazne čakalne vrste.
- **500 Mbps** – Prekomerno. Še nižja zakasnitev, a minimalna razlika v primerjavi s 150 Mbps.

**Optimalna kapaciteta:** Glede na razmerje med zmogljivostjo in stroški priporočamo **150 Mbps** kot optimalno kapaciteto povezav za dano konfiguracijo omrežja (7 vozlišč, 1 MB paketi, 1 s interval pošiljanja).

### 1.5.7 Povzetek rezultatov

Tabela 1.3: Primerjava rezultatov simulacij

Metrika	1 Mbps	30 Mbps	150 Mbps	500 Mbps
Povp. zakasnitev (s)	0	12.99	0.24	0.073
Izgubljeni paketi	0	61	109	109
Zasedenost vrste	50	25.4	0	0
Stanje omrežja	Blokirano	Obremenjeno	Optimalno	Prosto

## 1.6 Zaključek

V tem seminarju smo uspešno implementirali in analizirali krožno P2P topologijo v orodju OMNeT++. Glavni zaključki so:

1. **Hitrost prenosa** je najpomembnejši parameter, ki vpliva na zmogljivost krožnega omrežja.
2. Pri prenizki hitrosti (1 Mbps) omrežje postane popolnoma blokirano – čakalne vrste se napolnijo in paketi se ne morejo dostaviti.
3. Pri optimalni hitrosti (150 Mbps) dosežemo ravnotežje med nizko zakasnitvijo in učinkovito izrabo virov.
4. Krožna topologija ima inherentno slabost – paketi, ki ne dosežejo cilja, krožijo po omrežju in se vrnejo na izvor.

### 1.6.1 Možne nadgradnje

- Implementacija dvosmernega krožnega omrežja za krajše poti
- Dodajanje mehanizma za izbiro smeri pošiljanja
- Implementacija prioritetnih čakalnih vrst
- Analiza vpliva različnega števila vozlišč

## Literatura

1. W. source, “Orodje omnet++.” <http://www.omnetpp.org/>.